M4 Programmierprojekt

Rahmenbedingungen

- Arbeit allein oder zu zweit.
- Das Programmierprojekt zählt wie eine schriftliche Prüfung.

Ablauf

- Erstelle ein Github-Repository für das Projekt.
- Erstelle darin eine README.md-Datei, in die du zu Beginn erklärst, was das Projekt tut, warum es nützlich ist und wie es technisch umgesetzt werden soll (mindestens je ein aussagekräftiger Satz).
- Schicke die Repository-Adresse an chris.weber@kslzh.ch
- Regelmässiger Upload des Fortschritts auf Github (min. 1x pro Woche)
- Fertige Abgabe am Mittwoch, 21. Januar, 23:59. Bei der Bewertung wird berücksichtigt, dass Gruppe B mehr Zeit zur Verfügung hat.
- Kurze Präsentation und Vorführung des Projekts vor der Klasse.

Bewertungskriterien

Inhalt und technische Umsetzung

- Dein Projekt ist so ausgewählt und umgesetzt, dass du darin im EF-Unterricht Gelerntes korrekt und sinnvoll umsetzt und Sinnvolles dazulernst. Du findest eine zu deiner (Lern-)Erfahrung passende Balance zwischen Festigung/ Vertiefung und Erweiterung deiner Kenntnisse.
- Das Programm zeigt die gewünschte Funktion.
- Das Programm ist sinnvoll konzipiert und strukturiert.
- Die Benutzerführung ist intuitiv und wo nötig mit Bedienungshinweisen versehen, die erklären, was gerade passiert bzw. von der Benutzerin erwartet wird.
- Die Funktion und/ oder Umsetzung ist interessant und originell.

Kommunikation: Lesbarkeit, Dokumentation und Präsentation

- Der Code ist übersichtlich und verständlich:
  - Übersichtlicher Aufbau
  - Konsistente Einrückungen und Formatierungen (du kannst dich an den Beispielen aus dem Unterricht orientieren und die automatische Formatierung durch VSC nutzen).
  - Sprechende Bezeichnungen gemäss den Namenskonventionen (s.u.).
  - Kommentare, die den Code lesbarer machen. Auch die Kommentare sollten im Stil sowie der Verteilung eine gewisse Konsistenz aufweisen.

  - Freiwillige Vertiefungsmöglichkeit: Orientiere dich an einem gängigen Java Style Guide (bitte in der README-Datei angeben).
- Quellenangaben für Inspirationen und verwendeten Fremdcode (darf direkt in den Kommentaren sein).
- Die README-Datei enthält zuletzt alle laut Github nötigen Informationen: What the project does, Why the project is useful (das beinhaltet, warum es für dich sinnvoll/ lehrreich war), How users can get started with the project (d.h. kurze Bedienungsanleitung soweit nötig), Where users can get help with your project (ev. Verweis auf weitere Dokumentation), Who maintains and contributes to the project.[8] Die Datei kann einigermassen kurz gehalten sein, wenn das Programm gut kommentiert und die Benutzerführung gut ist.
- Die Präsentation vor der Klasse ist verständlich, interessant und unterhaltsam. Auf Fragen kann adäquat eingegangen werden.

Arbeitsprozess

- Einhalten der Termine und Rahmenbedingungen
- Sinnvolles Konzept
- Strukturiertes Vorgehen und Zeitmanagement, sinnvolle Zusammenarbeit innerhalb der Gruppe (falls zutreffend)
- Sinnvolles Testing. Die Tests sollen dokumentiert sein, z.B. indem sie mit kurzem Kommentar versehen in einer separaten Klasse abgelegt werden.

Namenskonventionen

- Alle Bezeichnungen sind sprachlich korrekt und in der gleichen Sprache (einer, die Mx Weber versteht).
- **Klassen**namen sind Substantive (da sie Vorlagen für Objekte sind) und beginnen mit einem Grossbuchstaben:
  `Fraction`, `String`, `Tester`, `FractionTester`
- **Methoden** tun etwas und haben deshalb Verbnamen (Camel Case: klein beginnen, danach jedes Wort gross):
  `getNumerator()`, `add()`, `move()`, `killAllOpenWindows()`
- **Variablen**namen sind auch in Camel Case:
  `variable`, `myVariable`, `numberOfElements`
- **Konstanten** werden durchgehend gross geschrieben und verwenden (als einzige) Unterstriche als Worttrenner:
  `PI`, `EULERS_NUMBER`, `EARTH_CIRCUMFERENCE`

[8] https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/about-readmes